Project1-hj2524_data_processing

Hanbo JIAO

2020/1/30

In this R notebook, we will process more on processed lyrics for our data analysis.

Step 0 - Load all the required libraries

From the packages' descriptions:

- `tm` is a framework for text mining applications within R;
- `data.table` is a package for fast aggregation of large data;
- `tidyverse` is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures;
- `tidytext` allows text mining using 'dplyr', 'ggplot2', and other tidy tools;
- `DT` provides an R interface to the JavaScript library DataTables.
- `wordcloud` is a package for plot wordcloud of data.
- `RColorBrewer`
- `reshape2`

```
packages.used=c("tm", "tidytext","tidyverse","DT","wordcloud","data.table","RColorBrewer","reshape2")

# check packages that need to be installed.
packages.needed=setdiff(packages.used,
                        intersect(installed.packages()[,1],
                                  packages.used))

# install additional packages
if(length(packages.needed)>0){
  install.packages(packages.needed, dependencies = TRUE)
}

# load packages
library(tm)
library(data.table)
library(tidytext)
library(tidyverse)
library(DT)
library(wordcloud)
library(RColorBrewer)
library(reshape2)

source("../lib/function_defined.R")
```

Step 1 - Load the data to be cleaned and processed

```
# load lyrics data from Text_processed
load('../output/processed_lyrics.RData')

# load artists information
artists <- read_csv("~/GitHub/Spring2020-Project1-hj2524/data/artists.csv")
```

```
## Parsed with column specification:
## cols(
##   Artist = col_character(),
##   Intro = col_character(),
##   Formed = col_double(),
##   Members = col_character(),
##   Origin = col_character()
## )
```

Step 2 - Determine whether artist is team or individual

```
# find a list of team artist
team_list<-artists%>%select(-Intro,-Origin)%>%filter(!is.na(Formed)|!is.na(Members))%>%filter(is.na(For

# add a column that indicates team or individual
data<-dt_lyrics%>%mutate(team=artist%in%team_list)%>%filter(year>1000)
```

Step 3 - Filter out Pop music only and Export data

```
# Filter out Pop music
data_Pop<-data%>%filter(genre%in% c("Pop"))

# Export data
save(data_Pop, file="../output/data_Pop.RData")
```

Step 4 - Count the frequent of positive & negative words in lyrics of Pop music based on sentiment "nrc" and Export data

```
# import sentiment nrc
nrc <- get_sentiments("nrc")

# combine sentiment with TDM of lyrics of pop music.
pop_nrc<-tdm(data_Pop)%>%inner_join(nrc,by=c("term"="word"))%>%rename("n"=`sum(count)`)

# Keep those positive and negative sentiments data.
pop_plot_sentiment<-pop_nrc %>%
  filter(sentiment%in%c("positive","negative"))%>%
  group_by(sentiment)

# Export data
save(pop_plot_sentiment, file="../output/pop_plot_sentiment.RData")
```

Step 5 - For different types of music, determine different types of artists are positive or negative based on sentiment "bing" and Export data

```
# import sentiment bing
bing<-get_sentiments("bing")
```

```r
# For any artists among all types of music,
# count the frequent of positive & negative words in lyrics of all of her songs,
# and find if there are more positive words or more negative words in all of her songs.
data_plot_sentiment<-
  data%>%group_by(genre,artist)%>%
  nest%>%mutate(tdm=map(data,tdm))%>%select(-data)%>%
  unnest(col=c("tdm"))%>%
  inner_join(bing,by=c("term"="word"))%>%rename("n"=`sum(count)`)%>%
  group_by(genre,artist,sentiment)%>%summarise(n=n())%>%
  spread(sentiment,n)%>%
  transmute(n=positive-negative)

# Export data
save(data_plot_sentiment, file="../output/data_plot_sentiment.RData")
```